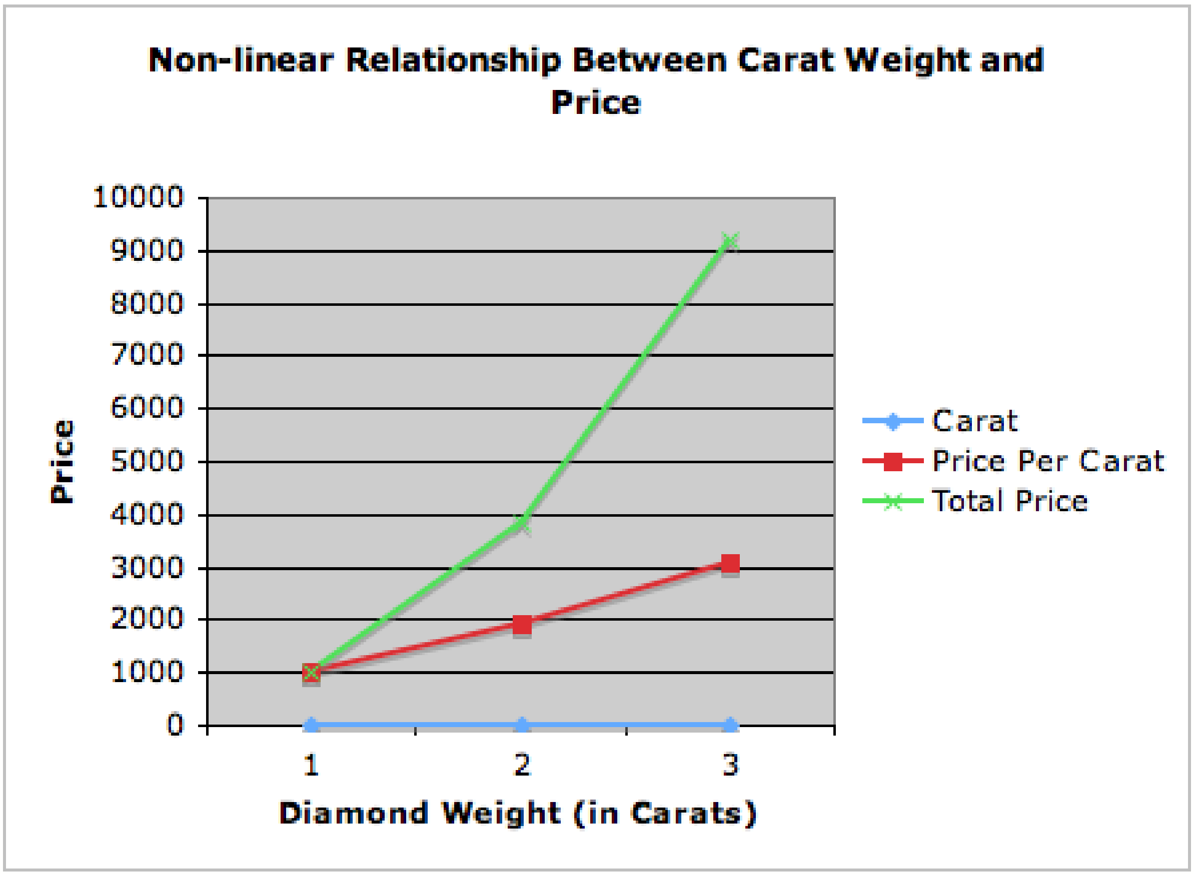
Non-linear Relationship Between Carat Weight and Price
Price
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
1
2
3
Diamond Weight (in Carats)
Carat
Price Per Carat
Total Price


Clarity Grades
Diamond Inhouse


AGS 0.5
AGS 2.0
AGS 3.5
AGS 5.0
AGS 8.0
Color Scale
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0 To Fancy Yellow
Colorless Near Colorless Faint Very Light Light Fancy Yellow
D E F G H I J K L M N O P Q R S T U V W X Y Z Fancy Yellow

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

(53940, 10)

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 53940 entries, 0 to 53939

Data columns (total 10 columns):

carat 53940 non-null float64

cut 53940 non-null object

color 53940 non-null object

clarity 53940 non-null object

depth 53940 non-null float64

table 53940 non-null float64

price 53940 non-null int64

x 53940 non-null float64

y 53940 non-null float64

z 53940 non-null float64

dtypes: float64(6), int64(1), object(3)

memory usage: 4.1+ MB

carat 0

cut 0

color 0

clarity 0

depth 0

table 0

price 0

x 0

y 0

z 0

dtype: int64

<matplotlib.axes._subplots.AxesSubplot at 0x1e5d54b59e8>

	carat	**depth**	**table**	**price**	**x**	**y**	**z**
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000
mean	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734
std	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699
min	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000
25%	0.400000	61.000000	56.000000	950.000000	4.710000	4.720000	2.910000

	carat	depth	table	price	x	y	z
50%	0.700000	61.800000	57.000000	2401.000000	5.700000	5.710000	3.530000
75%	1.040000	62.500000	59.000000	5324.250000	6.540000	6.540000	4.040000
max	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000

	carat	cut	color	clarity	depth	table	price	x	y	z
2207	1.00	Premium	G	SI2	59.1	59.0	3142	6.55	6.48	0.0
2314	1.01	Premium	H	I1	58.1	59.0	3167	6.66	6.60	0.0
4791	1.10	Premium	G	SI2	63.0	59.0	3696	6.50	6.47	0.0
5471	1.01	Premium	F	SI2	59.2	58.0	3837	6.50	6.47	0.0
10167	1.50	Good	G	I1	64.0	61.0	4731	7.15	7.04	0.0
11182	1.07	Ideal	F	SI2	61.6	56.0	4954	0.00	6.62	0.0
11963	1.00	Very Good	H	VS2	63.3	53.0	5139	0.00	0.00	0.0
13601	1.15	Ideal	G	VS2	59.2	56.0	5564	6.88	6.83	0.0
15951	1.14	Fair	G	VS1	57.5	67.0	6381	0.00	0.00	0.0

	carat	depth		table	price	x		y	z	
24394	2.18	Premium	H	SI2	59.4	61.0	12631	8.49	8.45	0.0
24520	1.56	Ideal	G	VS2	62.2	54.0	12800	0.00	0.00	0.0
26123	2.25	Premium	I	SI1	61.3	58.0	15397	8.52	8.42	0.0
26243	1.20	Premium	D	VVS1	62.1	59.0	15686	0.00	0.00	0.0
27112	2.20	Premium	H	SI1	61.2	59.0	17265	8.42	8.37	0.0
27429	2.25	Premium	H	SI2	62.8	59.0	18034	0.00	0.00	0.0
27503	2.02	Premium	H	VS2	62.7	53.0	18207	8.02	7.95	0.0
27739	2.80	Good	G	SI2	63.8	58.0	18788	8.90	8.85	0.0
49556	0.71	Good	F	SI2	64.1	60.0	2130	0.00	0.00	0.0
49557	0.71	Good	F	SI2	64.1	60.0	2130	0.00	0.00	0.0
51506	1.12	Premium	G	I1	60.4	59.0	2383	6.71	6.67	0.0

20

carat	cut	color	clarity	depth	table	price	x	y	z

<seaborn.axisgrid.FacetGrid at 0x1e5c1374128>

<matplotlib.axes._subplots.AxesSubplot at 0x1e5c1392fd0>

<matplotlib.axes._subplots.AxesSubplot at 0x1e5d7e4ebe0>

-
-
-
-
-
-
-
-
-
-
-
-
-
-

1.4) Drop the 'Unnamed: 0' column as we already have Index.

Great, So there are no NaN values.

Wait

- **Do you see the Min. Values of X, Y and Z. It can't be possible..!!**
- **It doesn't make any sense to have either of Length or Width or Height to be zero..**

Let's Have a look at them.

-

The Values are Distributed over a Small Scale.

-

Carat vs Price

-
-
-
-

<seaborn.axisgrid.JointGrid at 0x1e5c1392c18>

<seaborn.axisgrid.FacetGrid at 0x1e5cbf2a6d8>

<seaborn.axisgrid.FacetGrid at 0x1e5be64e438>

<seaborn.axisgrid.FacetGrid at 0x1e5da6cad30>

<seaborn.axisgrid.FacetGrid at 0x1e5cbd87358>

<matplotlib.axes._subplots.AxesSubplot at 0x1e5dbf17828>

```
(array([3.0000e+00, 0.0000e+00, 0.0000e+00, 0.0000e+00, 0.0000e+00,
        2.0000e+00, 4.0000e+00, 1.1000e+01, 4.3000e+01, 2.1900e+02,
        1.4240e+03, 5.0730e+03, 1.8242e+04, 2.2649e+04, 5.0330e+03,
        8.5100e+02, 2.3400e+02, 8.7000e+01, 2.7000e+01, 1.1000e+01,
        3.0000e+00, 1.0000e+00, 0.0000e+00, 0.0000e+00, 3.0000e+00]),
 array([43.  , 44.44, 45.88, 47.32, 48.76, 50.2 , 51.64, 53.08, 54.52,
        55.96, 57.4 , 58.84, 60.28, 61.72, 63.16, 64.6 , 66.04, 67.48,
        68.92, 70.36, 71.8 , 73.24, 74.68, 76.12, 77.56, 79.  ]),
```

<a list of 25 Patch objects>)

<seaborn.axisgrid.JointGrid at 0x1e5af1e9eb8>

<matplotlib.axes._subplots.AxesSubplot at 0x1e5da5e4da0>

<seaborn.axisgrid.JointGrid at 0x1e5db6a7e80>

(2, 10)

	carat	**cut**	**color**	**clarity**	**depth**	**table**	**price**	**x**	**y**	**z**	**volume**
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43	38.202030
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31	34.505856
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31	38.076885
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63	46.724580
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75	51.917250

(0, 50000)

<seaborn.axisgrid.JointGrid at 0x1e5dce49668>

####### Linear Regression #######

Score : 0.8814

[0.87116164 0.88350756 0.87757769 0.87635168 0.88384912]

MSE : 1911398.80

MAE : 926.72

RMSE : 1382.53

R2 : 0.88

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 0.0s finished

Lasso Regression

Score : 0.8659

[0.84325995 0.86900907 0.86386374 0.86539938 0.86976969]

MSE : 2162331.94

MAE : 909.60

RMSE : 1470.49

R2 : 0.87

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 0.1s finished

AdaBoost Regression

Score : 0.9093

[0.86364159 0.88625552 0.87610116 0.88472982 0.88632944]

MSE : 1462001.91

MAE : 827.10

RMSE : 1209.13

R2   : 0.91

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed: 17.9s finished

Ridge Regression

Score : 0.7537

[0.74232856 0.75599775 0.74753493 0.75626   0.74960313]

MSE   : 3970442.17

MAE   : 1346.18

RMSE   : 1992.60

R2   : 0.75

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed: 0.0s finished

Iter	Train Loss	Remaining Time
1	14009477.5296	0.95s
2	12437807.7359	0.72s
3	11113339.5845	0.75s
4	9945244.2308	0.76s
5	8973416.9156	0.78s
6	8109014.7842	0.79s
7	7387120.0500	0.78s
8	6753937.9878	0.77s
9	6197182.6819	0.76s
10	5724689.0901	0.76s
20	3200362.4597	0.68s
30	2393542.3170	0.58s
40	2102586.3335	0.49s
50	1923964.9187	0.41s
60	1790574.6006	0.32s

70 1688380.2826 0.24s

80 1609829.0076 0.16s

90 1548089.0039 0.08s

100 1499127.4566 0.00s

Iter Train Loss Remaining Time

1 13994442.1962 0.40s

2 12429322.7982 0.39s

3 11112606.0983 0.66s

4 9944843.0686 0.71s

...

MSE : 1518030.06

MAE : 720.72

RMSE : 1232.08

R2 : 0.91

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 4.0s finished

Random Forest

Score : 0.9809

[0.9782604 0.97855104 0.98081226 0.97523632 0.97956442]

MSE : 308285.07

MAE : 283.66

RMSE : 555.23

R2 : 0.98

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 9.9s finished

Score : 0.9822

R2   : 0.98

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed: 3.5s finished

KNeighbours Regression

Score : 0.9590

[0.95429058 0.95856983 0.95504994 0.94931403 0.95517559]

MSE   : 660416.40

MAE   : 424.98

RMSE   : 812.66

R2   : 0.96

Score : 0.9590

R2   : 0.96

	Algorithms	**R2-Scores**
5	RandomForest Regression	0.982167
6	KNeighbours Regression	0.959033
2	AdaBoost Regression	0.909309
4	GradientBoosting Regression	0.905833
0	Linear Regression	0.881432
1	Lasso Regression	0.865866
3	Ridge Regression	0.753705

<matplotlib.axes._subplots.AxesSubplot at 0x1e5af9edb70>

<seaborn.axisgrid.FacetGrid at 0x1e5af98c780>

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

(53940, 10)

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 53940 entries, 0 to 53939

Data columns (total 10 columns):

carat   53940 non-null float64

```
cut       53940 non-null object
color     53940 non-null object
clarity   53940 non-null object
depth     53940 non-null float64
table     53940 non-null float64
price     53940 non-null int64
x         53940 non-null float64
y         53940 non-null float64
z         53940 non-null float64
dtypes: float64(6), int64(1), object(3)
memory usage: 4.1+ MB
carat     0
cut       0
color     0
clarity   0
depth     0
table     0
price     0
x         0
y         0
z         0
dtype: int64
<matplotlib.axes._subplots.AxesSubplot at 0x1e5d54b59e8>
```

	carat	depth	table	price	x	y	z
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000
mean	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734
std	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699
min	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000
25%	0.400000	61.000000	56.000000	950.000000	4.710000	4.720000	2.910000
50%	0.700000	61.800000	57.000000	2401.000000	5.700000	5.710000	3.530000
75%	1.040000	62.500000	59.000000	5324.250000	6.540000	6.540000	4.040000
max	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000

	carat	cut	color	clarity	depth	table	price	x	y	z
2207	1.00	Premium	G	SI2	59.1	59.0	3142	6.55	6.48	0.0
2314	1.01	Premium	H	I1	58.1	59.0	3167	6.66	6.60	0.0
4791	1.10	Premium	G	SI2	63.0	59.0	3696	6.50	6.47	0.0
5471	1.01	Premium	F	SI2	59.2	58.0	3837	6.50	6.47	0.0

	carat	depth		table	price	x		y	z	
10167	1.50	Good	G	I1	64.0	61.0	4731	7.15	7.04	0.0
11182	1.07	Ideal	F	SI2	61.6	56.0	4954	0.00	6.62	0.0
11963	1.00	Very Good	H	VS2	63.3	53.0	5139	0.00	0.00	0.0
13601	1.15	Ideal	G	VS2	59.2	56.0	5564	6.88	6.83	0.0
15951	1.14	Fair	G	VS1	57.5	67.0	6381	0.00	0.00	0.0
24394	2.18	Premium	H	SI2	59.4	61.0	12631	8.49	8.45	0.0
24520	1.56	Ideal	G	VS2	62.2	54.0	12800	0.00	0.00	0.0
26123	2.25	Premium	I	SI1	61.3	58.0	15397	8.52	8.42	0.0
26243	1.20	Premium	D	VVS1	62.1	59.0	15686	0.00	0.00	0.0
27112	2.20	Premium	H	SI1	61.2	59.0	17265	8.42	8.37	0.0
27429	2.25	Premium	H	SI2	62.8	59.0	18034	0.00	0.00	0.0
27503	2.02	Premium	H	VS2	62.7	53.0	18207	8.02	7.95	0.0
27739	2.80	Good	G	SI2	63.8	58.0	18788	8.90	8.85	0.0

	carat	depth		table	price	x		y	z	
49556	0.71	Good	F	SI2	64.1	60.0	2130	0.00	0.00	0.0
49557	0.71	Good	F	SI2	64.1	60.0	2130	0.00	0.00	0.0
51506	1.12	Premium	G	I1	60.4	59.0	2383	6.71	6.67	0.0

20

	carat	cut	color	clarity	depth	table	price	x	y	z

<seaborn.axisgrid.FacetGrid at 0x1e5c1374128>

<matplotlib.axes._subplots.AxesSubplot at 0x1e5c1392fd0>

<matplotlib.axes._subplots.AxesSubplot at 0x1e5d7e4ebe0>

-
-
-
-
-
-
-
-
-
-
-
-

-
-

1.4) Drop the 'Unnamed: 0' column as we already have Index.

Great, So there are no NaN values.

Wait

- **Do you see the Min. Values of X, Y and Z. It can't be possible..!!**
- **It doesn't make any sense to have either of Length or Width or Height to be zero..**

Let's Have a look at them.

-

The Values are Distributed over a Small Scale.

-
-
-
-
-
-

Carat vs Price

-
-
-

-
-
-

<seaborn.axisgrid.JointGrid at 0x1e5c1392c18>

<seaborn.axisgrid.FacetGrid at 0x1e5cbf2a6d8>

<seaborn.axisgrid.FacetGrid at 0x1e5be64e438>

<seaborn.axisgrid.FacetGrid at 0x1e5da6cad30>

<seaborn.axisgrid.FacetGrid at 0x1e5cbd87358>

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e5dbf17828>
```

```
(array([3.0000e+00, 0.0000e+00, 0.0000e+00, 0.0000e+00, 0.0000e+00,
        2.0000e+00, 4.0000e+00, 1.1000e+01, 4.3000e+01, 2.1900e+02,
        1.4240e+03, 5.0730e+03, 1.8242e+04, 2.2649e+04, 5.0330e+03,
        8.5100e+02, 2.3400e+02, 8.7000e+01, 2.7000e+01, 1.1000e+01,
        3.0000e+00, 1.0000e+00, 0.0000e+00, 0.0000e+00, 3.0000e+00]),
 array([43.  , 44.44, 45.88, 47.32, 48.76, 50.2 , 51.64, 53.08, 54.52,
        55.96, 57.4 , 58.84, 60.28, 61.72, 63.16, 64.6 , 66.04, 67.48,
        68.92, 70.36, 71.8 , 73.24, 74.68, 76.12, 77.56, 79.  ]),
 <a list of 25 Patch objects>)
```

```
<seaborn.axisgrid.JointGrid at 0x1e5af1e9eb8>
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e5da5e4da0>
```

```
<seaborn.axisgrid.JointGrid at 0x1e5db6a7e80>
```

```
(2, 10)
```

	carat	cut	color	clarity	depth	table	price	x	y	z	volume
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43	38.202030
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31	34.505856
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31	38.076885
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63	46.724580
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75	51.917250

(0, 50000)

<seaborn.axisgrid.JointGrid at 0x1e5dce49668>

####### Linear Regression #######

Score : 0.8814

[0.87116164 0.88350756 0.87757769 0.87635168 0.88384912]

MSE   : 1911398.80

MAE   : 926.72

RMSE   : 1382.53

R2   : 0.88

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed: 0.0s finished

Lasso Regression

Score : 0.8659

[0.84325995 0.86900907 0.86386374 0.86539938 0.86976969]

MSE : 2162331.94

MAE : 909.60

RMSE : 1470.49

R2 : 0.87

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 0.1s finished

AdaBoost Regression

Score : 0.9093

[0.86364159 0.88625552 0.87610116 0.88472982 0.88632944]

MSE : 1462001.91

MAE : 827.10

RMSE : 1209.13

R2 : 0.91

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 17.9s finished

Ridge Regression

Score : 0.7537

[0.74232856 0.75599775 0.74753493 0.75626 0.74960313]

MSE : 3970442.17

MAE : 1346.18

RMSE : 1992.60

R2 : 0.75

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 0.0s finished

```
      Iter       Train Loss      Remaining Time
         1      14009477.5296           0.95s
         2      12437807.7359           0.72s
         3      11113339.5845           0.75s
         4       9945244.2308           0.76s
         5       8973416.9156           0.78s
         6       8109014.7842           0.79s
         7       7387120.0500           0.78s
         8       6753937.9878           0.77s
         9       6197182.6819           0.76s
        10       5724689.0901           0.76s
        20       3200362.4597           0.68s
        30       2393542.3170           0.58s
        40       2102586.3335           0.49s
        50       1923964.9187           0.41s
        60       1790574.6006           0.32s
        70       1688380.2826           0.24s
        80       1609829.0076           0.16s
        90       1548089.0039           0.08s
       100       1499127.4566           0.00s
      Iter       Train Loss      Remaining Time
         1      13994442.1962           0.40s
         2      12429322.7982           0.39s
         3      11112606.0983           0.66s
         4       9944843.0686           0.71s
...
MSE    : 1518030.06
MAE    : 720.72
```

RMSE : 1232.08

R2 : 0.91

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 4.0s finished

Random Forest

Score : 0.9809

[0.9782604 0.97855104 0.98081226 0.97523632 0.97956442]

MSE : 308285.07

MAE : 283.66

RMSE : 555.23

R2 : 0.98

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 9.9s finished

Score : 0.9822

R2 : 0.98

[Parallel(n_jobs=1)]: Done 5 out of 5 | elapsed: 3.5s finished

KNeighbours Regression

Score : 0.9590

[0.95429058 0.95856983 0.95504994 0.94931403 0.95517559]

MSE : 660416.40

MAE : 424.98

RMSE : 812.66

R2 : 0.96

Score : 0.9590

R2 : 0.96

	Algorithms	R2-Scores
5	RandomForest Regression	0.982167
6	KNeighbours Regression	0.959033
2	AdaBoost Regression	0.909309
4	GradientBoosting Regression	0.905833
0	Linear Regression	0.881432
1	Lasso Regression	0.865866
3	Ridge Regression	0.753705

<matplotlib.axes._subplots.AxesSubplot at 0x1e5af9edb70>

<seaborn.axisgrid.FacetGrid at 0x1e5af98c780>